Capstone Proposal Andrea Balzano

Machine Learning Engineer Nanodegree 20th April 2017

*The Search for New Earths*

Definition

Project overview

This project analyses the data observed by the NASA Kepler space telescope searching for exoplanets using the transit technique.

planets themselves do not emit light, but the stars that they orbit do. If said star is watched over several months or years, there may be a regular 'dimming' of the flux (the light intensity). This is evidence that there may be an orbiting body around the star; such a star could be considered to be a 'candidate' system.

NASA itself utilises python to interpret the data and has created PyKE, a library for data reduction to help with extraction and preprocessing of the light curve images, however this project analyses only FLUX data, not pictures.

Some of the machine learning techniques already been used by developers are 1-D CNN, XGBoosting, PCA.

Known Dataset problem

The Transit technique has some flows, it can be used to investigate only stars lying on the same hyperplane as the sun and it can detect only planets that orbits their stars with a very short period because it needs to be able to see more than one orbit to be able to differentiate between a transit and background noise.

For example, looking at our own solar system, to be able to confirm the existence of the Earth we would have to constantly observe the sun for at least 2 years to see 2 transits and it probably wouldn’t be enough information to differentiate the dip in light caused by the Earth because of the possibility of dips caused other bodies.

To identify Jupiter Following this technique we would have to observe the sun for 24 years just to see 2 transits and for some outer planets like Saturn and Pluto respectively 59 and 496 years.

Source: [Nasa](https://www.nasa.gov)

For this project, it means that observation marked as non-exoplanet systems, might have planets, creating extra noise and further complicating the classification task.

Problem Statement

The goal is to create an agent able to classify candidate systems.

At the time this dataset was prepared Campaign-3 was unlikely to contain any undiscovered exoplanet-stars. Therefore, all stars (i.e. all rows) which were not confirmed to host an exoplanet, were labelled with a 0. This is over 99% of the observations.

In total, there are 42 observations of confirmed exoplanets labelled with a 1, 5 in the test set and 37 in the train set.

The task involved are the following:

1 Download the dataset.

2 remove outliers and normalise the data.

3 correct for class imbalance

4 run preprocessing analysis.

5 run classification algorithm.

The output of the classifier is a binary class where a system not containing exoplanets is labelled 0, and a candidate for more investigations is labelled 1.

Metrics

class 0 includes 99% of samples, due to this high class label imbalance, the metric used will be the sensitivity and specificity scores.

Sensitivity (also called the true positive rate) measures the proportion of positives that are correctly identified as such.

i.e. the percentage of candidate systems which are correctly identified as candidate.

Specificity (also called the true negative rate) measures the proportion of negatives that are correctly identified as such.

i.e. the percentage of systems which are correctly identified as not being candidate.

This metric was used to evaluate the classifier because it best represents false negative and false positive when high class imbalance.

* False negatives result in a missed opportunity, these are systems that actually have exoplanets but are not classified as such.
* False positives result in further investment of resources to study in details the system only to find out that we can’t identify any exoplanets.

Analysis

Data exploration

the Trainset includes:

• 5087 rows or observations.

• 3198 columns or features.

• Column 1 is the label vector. Columns 2 - 3198 are the flux values over time.

• 37 confirmed exoplanet-stars and 5050 non-exoplanet-stars.

The Testset includes:

• 570 rows or observations.

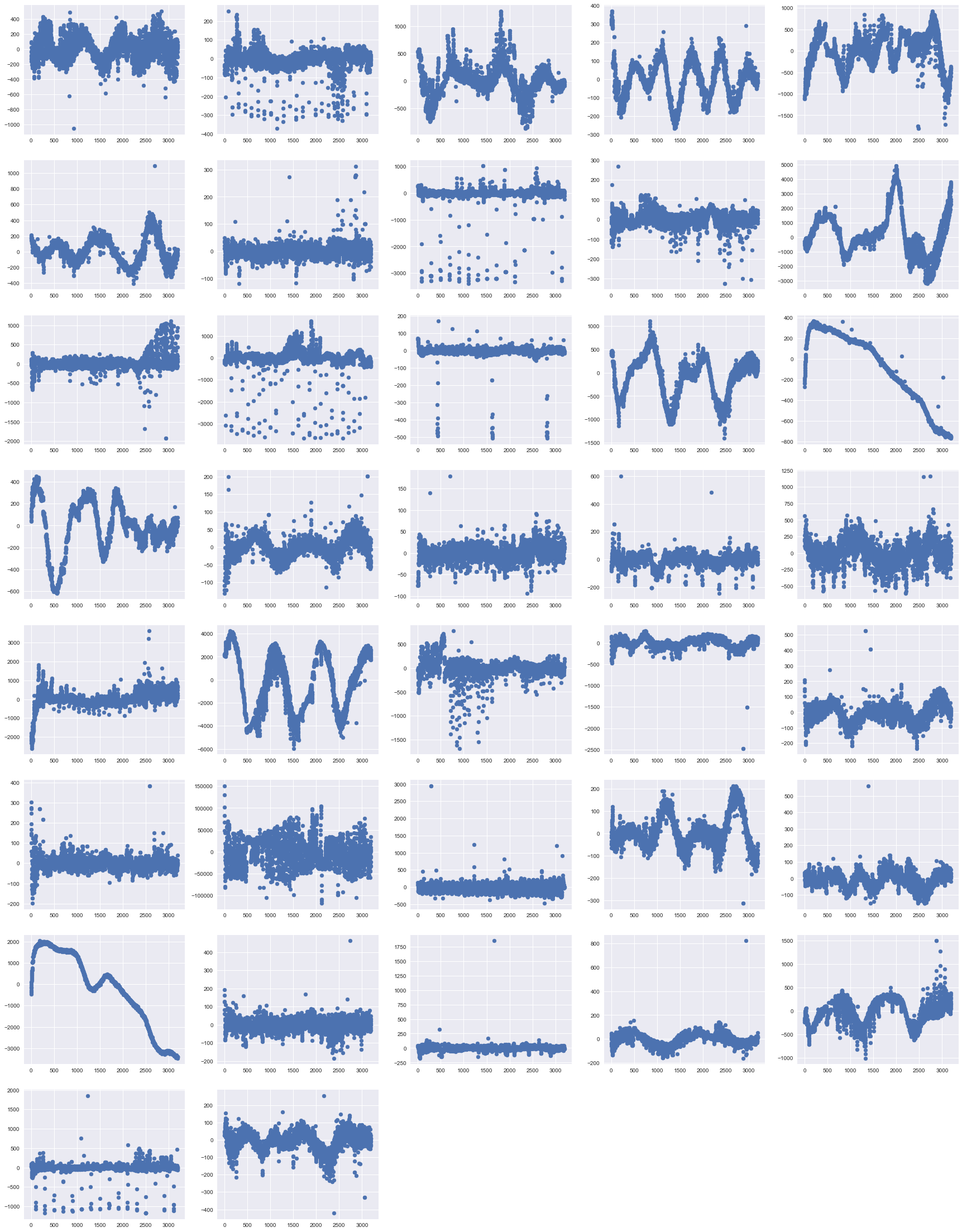
• 3198 columns or features.

• Column 1 is the label vector. Columns 2 - 3198 are the flux values over time.

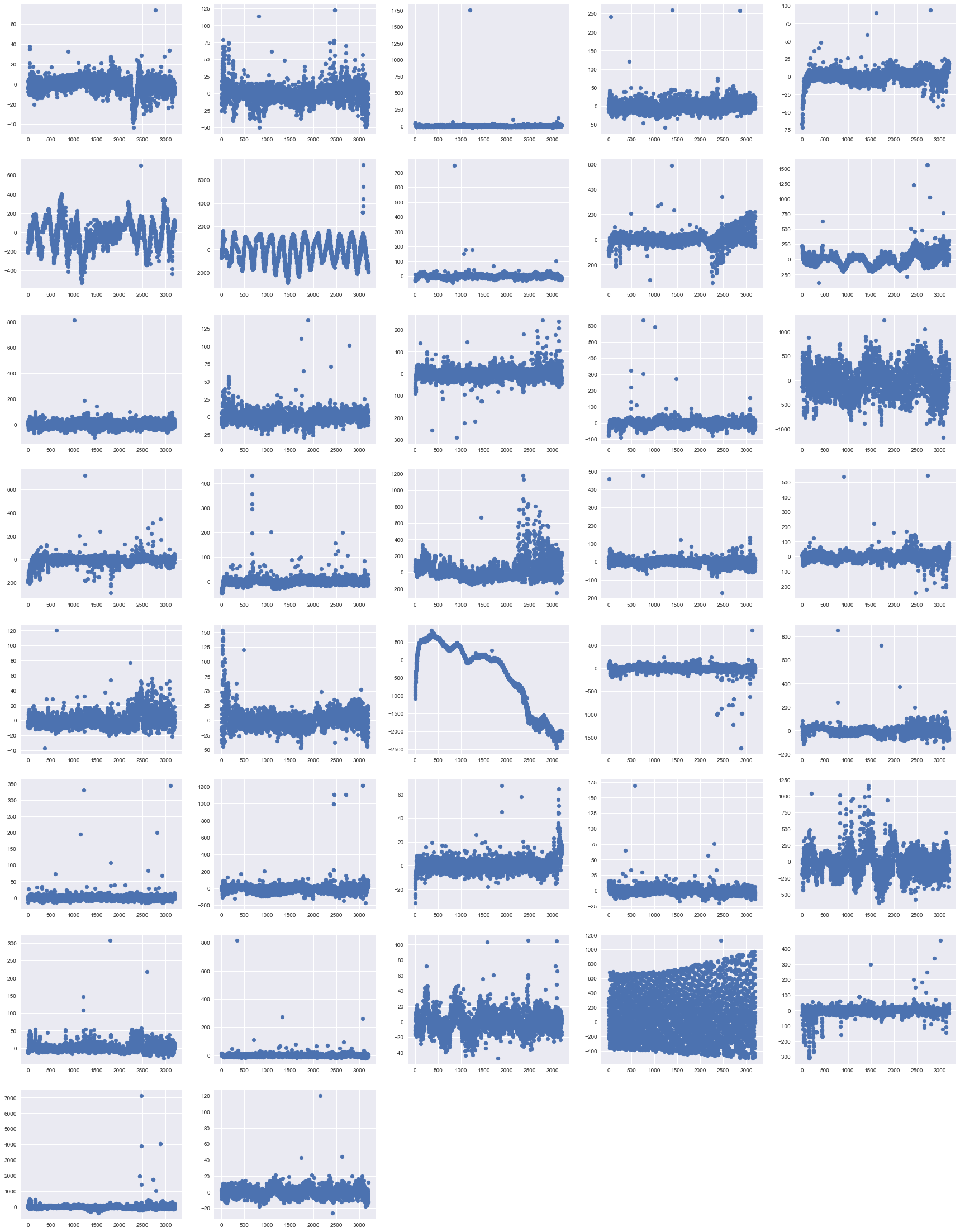
• 5 confirmed exoplanet-stars and 565 non-exoplanet-stars.

Exploratory Visualisation

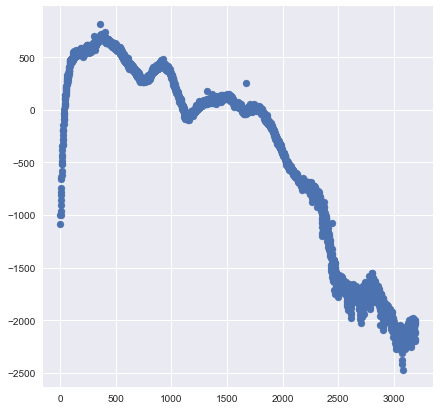
**Fig 1**: Shows how the luminescence dims when an exoplanet transit in front of the star for the 37 known exoplanets in the trainingset, in these cases we can observe system with multiple, short, or long transits:



**Fig 2:** shows outliers for class 0,non-exoplanets systems, some of these observation presents strong sinusoidal noise or high fluctuations in short times that might be caused by interference with instruments, others might actually show one transit of a planet that we are not yet able to identity.



**Fig 3**: this particular outlier could be showing the transit of a slow-moving gas giant planet.



Algorithms and Techniques

I explored multiple techniques in this project, some more successful then others.

Initially considering that This dataset has very high dimensionality represented by the time series of the FLUX measurements, I did analyse the data looking for outliers that can be due to noise or sensors anomalies, approaching the detection using Tukey's Method for identifying outliers, where an outlier step is calculated as 1.5 times the interquartile range (IQR).

I Choose his method because it isn’t dependent on distributional assumptions, and It also ignores the mean and standard deviation, making it resistant to being influenced by the extreme values in the range.

I filtered all the stars with measurements under/over the outlier step looking for periodicity on these values to try isolate noise and decide on a case basis if to keep the record, remove it or replace the value of the outlier with the median value for that column.

I addressed the class imbalance by applying SMOTE + Tomek links to try to balance the tradeoff of over/undersampling.

The SMOTE technique increases the size of the minority class by creating samples in the neighbours, correcting for the tendency to overfit to the samples common during oversampling.

The Tomek links is a redundancy driven technique to reduce sampling size, it looks for Tomek-links which consist of points that are each other’s closest neighbours, but do not share the same class label.

For a binary class, the number of possible combinations grows exponentially with the number of dimensions, 2^n where n is the number of features, this is also known as the curse of dimensionality.

I did apply features transformation with PCA to capture the delta changes over time, reducing the number of features and compensating for the curse of dimensionality.

I did try Gaussian Naïve Bayes, SVM, XGboosting classifiers to get a benchmark, and while all classifier tested had a strong tendency to overfit, I explored the latest algorithm to fine tune it using cross-validation on the training set.

Ones the score on the validation set was acceptable I did evaluate the performance on the testset.

I noticed that without normalizing the data PCA was very effective in reducing dimensionality while retaining information, with 2 features it could explain 83% of variance and 99% with just 10.

After applying normalization, normalizing by sample or using standard scaler the PCA was little effective and I had to use 30 - 50 features to explain just 45% of the variance.

The results of the test set following this solution where always biased towards one of the two classes, the best classifier had about a sensitivity of 80% and specificity of 7%.

I investigated the possibility of doing a time series analysis but the algorithm are designed for problems with variable outcome overtime while this problem is about an overtime change that leads to one outcome.

Since then I moved to one class classifier for outlier detection, I approached the problem differently by not applying outlier removal or class imbalance correction, training only on class 0.

The algorithm I used are one class SVM and isolation forest.

Benchmark

This project could be defined as 'A Search for a Needle in a Haystack' therefore I wouldn’t expect an algorithm that can classify systems with high accuracy, I will consider a precision on class 1 of 30 % a good benchmark for this project.

Project Design

1. Outliers detection using Tukey's Method for identifying outliers.
2. Class imbalance correction by applying SMOTE + Tomek links.
3. Features transformation with PCA.
4. Creation of a benchmark classifier, I will implement a simple classifier in naïve Bayes, SVM, decision tree and use the best model as benchmark.
5. Develop a Neural Network classifier using Tensorflow.
6. Cross validation on the training set as validation.
7. Classify testset.